Practical Protection for Personal Storage in the Cloud

Neal H. Walfield, Paul T. Stanton, John Linwood Griffin and Randal Burns
Johns Hopkins University

EuroSec ’10
April 13th, 2010
Outline

- Personal Storage Today
- Practical Protection Mechanisms
Web 2.0: Today

- Each service provides the user with storage
- Limited support for sharing between services
An Emerging Issue

- **Data Management is Hard!**
  - Data Lock-In
    - No standardized access interface (à la POSIX)
    - Must use service’s interface; point solutions
  - Data Spew
    - Data is hard to find
  - Version Drift
    - Sharing across services $\Rightarrow$ divergent copies
An Emerging Issue

- **Data Management is Hard!**
  - Data Lock-In
    - No standardized access interface (à la POSIX)
    - Must use service’s interface; point solutions
  - Data Spew
    - Data is hard to find
  - Version Drift
    - Sharing across services $\implies$ divergent copies

- Underlying Architectural Problem:
  - **Many storage providers**
  - $\implies$ No unified view of data
A Simple Solution: One Storage Provider

- User has direct access to data
- Single, authoritative copy
- Cross-service sharing
A Simple Difficulty

- **Access Control**
  - Facebook should not be able to access EMail
A Simple Difficulty

- **Access Control**
  - Facebook should not be able to access EMail

- **Reputation!**
A Simple Difficulty

- **Access Control**
  - Facebook should not be able to access EMail

- **Reputation is not enough!**
  - Users less likely to experiment
  - Raises barrier to entry
Outline

- Personal Storage Today
- Practical Protection Mechanisms
Per-User Storage: Major Design Goals

- Protection
  - Least Privilege
    - Not Unix
    - Fine-grained, dynamic delegation and revocation

- Usability
  - Minimal user interactions with security manager
    - Opening, saving files
  - Delegate access to not-yet-existing objects
    - Flickr can access all JPEG files
  - Consistent naming of objects
    - `/photos/paris/dsc_1076.jpg` always has same name
S4: Simple, Secure Storage Service

- Hierarchical Principals
- Filtered Views
- Powerbox
  - Security manager implements open, save-as dialogs
Principals

Hierarchical
  - Alice dominates Alice.Hotmail

Principals identified using public key cryptography
Creating a new Principal

▷ Credentials communicated using a Webkey
  ▷ Includes service’s public, private keys
  ▷ Includes storage server’s public key
Filtered Views

- Filter parent’s name space
  - Principal can access that which it can name
- e.g., Regular expressions
- Enables consistent naming, future delegations

Alice:
- rw, /addressbook
- rw, /Maildir

Alice.Hotmail:
- rw, /addressbook

Alice.Facebook:
- rw, /addressbook
Filtered Views

- Filter parent’s name space
  - Principal can access that which it can name
- e.g., Regular expressions
- Enables consistent naming, future delegations

Alice

rw, /addressbook
rw, /Maildir

Alice.Hotmail

Alice.Facebook

rw, /addressbook
/Maildir/...
/photos/...
/calendar/...
...

Filter parent’s name space
- Principal can access that which it can name
- e.g., Regular expressions
- Enables consistent naming, future delegations
Powerbox

Least Privilege View  Powerbox View
Powerbox

Concept

- Replaces application’s open, save-as dialog box
- Service sends an RPC to security manager
- Security manager displays dialog box

Essential for usable least privilege

- Dynamic delegation
- No (explicit) user interactions with security manager
Integrating the Powerbox into Flickr

- Alice creates a Flickr account at flickr.com
- Alice creates a principal using security manager
- Alice gives credentials to Flickr
- Flickr starts an import photos wizard
  - Invokes Powerbox
  - What files would you like to import to Flickr?
  - Alice selects one or more directories
Integrating the Powerbox into Flickr

- Alice creates a Flickr account at flickr.com
- Alice creates a principal using security manager
- Alice gives credentials to Flickr
- Flickr starts an import photos wizard
  - Invokes Powerbox
  - What files would you like to import to Flickr?
  - Alice selects one or more directories

- Differences:
  - One additional step
  - But, Alice can use her own tools to upload photos
Powerbox Protocol in S4

1. File $\rightarrow$ Open
2. pb_invoke
3. Open Dialog
4. delegate, pb_close
5. pb_close
Performance

- User’s storage is authoritative
- Services can (should) still cache
  - Prompt propagation of updates
Adoption

- User’s want it
  - Improved usability, control
  - Current services lost control
    - Differentiator for new service providers
Adoption

- User’s want it
  - Improved usability, control
  - Current services lost control
    - Differentiator for new service providers

- Big services providers want it?
  - Increase user traffic by becoming a storage provider
Implementation

- 4000 lines of Python (SLOCCount)
  - Single machine, Single threaded
- S3 compatible
- S3 and SQLite backends
- Principal and filter interfaces complete, some Powerbox
Future Work

- Filters based on files’ tags
- Snapshots for recovery
- COW for experimentation
- Publish/subscribe for updates
- Throttling bandwidth intensive services
- Do not disclose content to server
Summary

The Bad (the status quo)

- Data lock-in
- Data spew
- Version drift

The Good (what S4 tries to achieve)

- Single (perceived) file system
- Least privilege
- Minimal user interaction with security monitor
  - Powerbox
  - Protection mechanisms consistent with user’s intuitions
    - All JPEG files
- Delegate access to not-yet-existing objects
- Consistent naming of objects
Take Aways

- Filtering matches how users think about security policies
- Powerbox helps make security invisible
Image Attributions

- User Images - User Experience Deliverables by Peter Morville and Jeffery Callender - http://www.flickr.com/photos/morville/3220961846/ - CC Attribution 2.0


Summary

The Bad (the status quo)
- Data lock-in
- Data spew
- Version drift

The Good (what S4 tries to achieve)
- Single (perceived) file system
- Least privilege
- Minimal user interaction with security monitor
  - Powerbox
  - Protection mechanisms consistent with user’s intuitions
    - All JPEG files
- Delegate access to not-yet-existing objects
- Consistent naming of objects